FINAL VERSION OF REPLACEMENT PAGES OF SPECIFICATION

FIG. 1 is a block diagram illustrating one embodiment of a dialog computer system according to the present invention. The various parts of the system can be implemented on a single computer system with appropriate programming. It could be implemented in any number of computer programming languages, including Java or other object-oriented type programming languages. The process of parsing a user's phrase and selecting an appropriate response can be divided into several major operations, that are shown in FIG.1 as separate modules. The operation of the system will be described in conjunction with the processes in each of these modules.

To implement the process, a session object is created and stored in the short-term memory device 4. The session object links all processes in the system and defines a unique current contact for every user, including a list of topic instances. A static cache list is maintained to prevent the creation of redundant topic instances. This is important because the topics do not contain session specific data. The content of the static cache list is determined by an initialization file of the system indicating which pattern knowledge bases should be available to the mechanisms of the system during a conversation.

First, the preprocessor 12 receives and prepares a user's natural language phrase 00 to be processed by the other modules of the system. The user's phrase can be received in a variety of ways, including inputted on a keyboard or through voice recognition. Additionally, the phrase may be of some form of constrained natural language, such as brief phrases, menu-type phrases, and commands. For purposes of the embodiment of the present invention, it is sufficient if the user's phrase is any sequence of words or symbols reflecting a user's intended statement or response. In general, the pre-processor 12 prepares a user's phrase for the next module - pattern recognition mechanism 13. The preprocessor module 12 removes extraneous information, such as special, unconventional cases and special non-dividable phrases and prefixes. Items such as titles and URL addresses are processed and translated into a form that can be understood by the pattern recognition mechanism 13.

The processes performed in the preprocessor module 12 are illustrated in Fig. 2. First, the preprocessor breaks the user phrase into the units of the text, separated by spaces and characters such as "[", "]", "(", ")" (step 32). At step 34, the preprocessor 12 prevents the separation of those pieces of the text that should not be changed, for example URL addresses, titles (like "Mr.", "i.e."), if necessary. At steps 35-38, the preprocessor 12 removes those parts of the text that are not keywords in the pattern. Non-keywords may include words or punctuation marks. The remaining part of the user's phrase is separated into parts to be parsed (step 39). The parts can be separated by special characters, such as ".", "-",

such as ".", "-", ">", etc. Finally, the preprocessor may implement any user-defined functions (step 40). The ability to add user-defined functions increases the personalization and accommodates the idiosyncrasies of each user. Once the user's phrase has been preprocessed, it is passed to the pattern recognition module 13.

The pattern recognition mechanism 13 identifies important elements in the input phrase and looks for the best matches between these elements and the patterns in the pattern knowledge bases 1-3, using special expression-pattern matching techniques. It then retrieves user input objects linked with the selected patterns from the pattern knowledge bases 1-3. The user input objects define the topic of the user's phrase, the intention conveyed by the user utterance and topic properties, which describe the content of the phrase as understood by the pattern recognition module 13. The pattern matching process is generally illustrated in Fig. 4. The function can be separated into two main parts: match data building 45 and matching with particular match data 46. A spell checking process 47 can be used as part of the particular data matching 46.

FIGS. 5-7 illustrate the processes for initialization of the pattern recognition mechanism 13 from the pattern information in the knowledge bases 1-3. At the stage of initialization (step 51), the pattern recognition mechanism 13 uses the patterns in the knowledge bases 1-3 to create pattern trees. A default dictionary 55 is created by using the customer's dictionaries or general dictionaries for the system. The pattern trees include a subject detector tree 53, a default tree 52, a token tree 54, and any subject specific trees. The processes for creating trees are illustrated in Figs. 6 and 7. The subject detector tree 53 is composed of the existent subjects. Each subject has a separate specific tree. A pattern is checked (step 57) to determine whether it has a specific subject. If the pattern has its own defined subject it is entered in the subject specific trees (step 58). All patterns, with or without a subject, are also entered into the default tree (step 59). The process for entering a pattern on a tree is illustrated in Fig. 7. The pattern is parsed into a sequence of individual nodes (step 62). The nodes are added to the tree in order (step 63). After entering either of the trees, the patterns with similar beginnings share a common path in the tree structure, taking different branches at the points where they differ. For example, Fig. 3 illustrates a tree structure for the following set of patterns:

* chatterbot *
When will * ready
When * chatterbot *

Using the user or default dictionary 55 and predefined keyboard maps 67, the spell checker builds a numeric equivalent of the user phrase. The spelling checker receives a word from the preprocessed user's phrase and checks if the word is included in the dictionary (70). If the word is in the dictionary, its numeric value is retrieved (step 75). Otherwise, it checks whether the word consists of two correct words from the dictionary 55 with the missing space between them (step 71). If it is true, then it inserts a space and proceeds to retrieve the numeric value for the words (step 75).

At this stage the normalization of the word including recognizing its suffix and checking the correctness of the word remaining after its suffix is detached can be done. Usually, word normalization is used for decreasing the number of words in a dictionary and for increasing the speed of processing of the whole phrase in the spelling checker 47, respectively. Otherwise it defines the "nearest" words in the dictionary and replaces the incorrect word by the word the spelling of which is the closest to it (steps 72-74). To define this word the spelling checker 47 calculates the distance between the user word and each word from the dictionary by means of a special set of numbers which is used for defining weights of letters according to the position of the keys on a keyboard. The set of numbers is presented below only for the English keyboard map. The English language is selected as natural language only for illustrative purpose. However, it will be understood by anyone reasonably skilled in the arts that any language may be used as long as appropriate keyboard maps are used.

Table 1. Letter's weights used for spelling- check

Letter	Row weight	Column weight	Letter	Row weight	Column weight	Letter	Row weight	Column weight
A	2	1	J	2	7	S	2	2
B	3	5	K	2	8	T	1	5
C	3	3	L	2	9	U	1	6
D	2	3	M	3	7	V	3	4
E	1	3	N	3	6	W	1	2
F	2	4	O	1	9	X	3	2
G	2	5	P	1	10	Y	1	6
H	2	6	Q	1	1	Z	3	1
I	1	8	R	1	4	‘	2	11

system has just been asked a question, then a subsequent input from the user cannot be marked as "reply to a question". Similarly, if the system has just asked a question, then the topic-centric arrangement of the patterns in the knowledge base gives the meta-control mechanism the clue it needs to recognize if the user gives a response that might be very generic - *outside* the context, but specific for a particular situation. For instance, the word "yes" is virtually meaningless without a context, but as an answer to a specific question it is extremely important if it matches the context of that question. Thus, the mechanism can recognize situations of the type and hence filter out untenable hypotheses, keeping only the most appropriate candidates.

Of course, a context-related question is not the only issue involved in making an intelligent conversation, and in many cases an adequate context is not easily available. In such cases, the meta-control mechanism 14 can generate a default assumption, which is used for conducting a conversation, based upon the synthesis of its own fundamental dialogue rules and a special analysis carried out by the discourse control mechanism 15, the goal control mechanism 16, the emotion control mechanism 17. To implement its own tasks the meta-control mechanism uses a strategy of filtering the recognized and parsed user inputs with applying the conversation knowledge described by the meta-rules taken from the corresponding base, as well as a strategy of choosing the best user input. Besides, it uses a method of processing the information received from the other called mechanisms to define a system output and meta-actions. The process describing the stages of processing the information received from the pattern recognition mechanism 13 is illustrated in FIG. 10.

At the first step (steps 110-113) the set of patterns from the pattern recognition module 13 are preprocessed to eliminate extraneous possibilities. The possible patterns are sorted according to the type of topics and each user input from the set is filtered with the help of the meta-rules describing the conversation knowledge. It allows removing inputs which are not applied to a given case.

For example, the following list contains meta-filtering rules with their own dialogue-based filtering rules. These rules relate, in part, to an intention type for each phrase. The intention types relate to the expected responses from a user. For example, a REQUEST type means that the user is asking a question or seeking information. A MORE_REQUEST type means that the user is seeking additional information on the same subject. A REPLY type means that the user is responding to a system question. The system responses are referred to as scripts. Scripts also have types, such as REQUEST, STATEMENT (which provides

corresponding to the selected user input (step 130). Such sequence of actions allows choosing the best topic to be further discussed as well as avoiding a situation when the system does not fully understand a user phrase and there is a need to change the current topic for another one in a more human-like and reasonable way.

For instance, when the system in the form of a tutor-application and a student are involved in an intensive lesson, and suddenly the student makes a remark that is not germane to the lesson, the discourse control mechanism 15 of the considered system applies a generic model of behavior that enables the system to respond to the remark in the right manner, and then steer the conversation back to the topic. To implement this process a knowledge engineer only needs to worry about compiling the knowledge received and working out a teaching strategy which is needed for the subject matter, and does not need to worry about predicting all contingencies that can happen during a conversation.

Once a user input is determined by the meta-control mechanism 14, using appropriate discourse control sub-mechanisms, the system selects and prepares a response to the input. The meta-control mechanism 14 generates the best possible script (or response) characterized by a set of properties, taking into account the user's state (represented by the user input object), the memory, the state of the system (represented by the emotional state controlled by the emotional control mechanism 17 and the state of the goals determined by the goal control mechanism 16) and a list of the best possible answers presented by the discourse control mechanism 15 and selected by considering the correspondence of the involved emotional contents and some other rules. All possible system answers to the best user input 140 selected at the previous stage are chosen from the associated actions base 11. This choice is based on the correspondence of the attributes of the chosen input and scripts including a set of pre-defined actions, intentions of the considered input, script and the type of the activated discourse control sub-mechanism. To make the above choice the set forth below rules with pre-defined descending priorities is applied (Fig. 11, steps 141, 142).

1. If the OBJECT-type discourse control sub-mechanism is activated and user input has a REPLY-type intention or an intention which shows that the user enters some information into the system then the following outputs are generated:

- A REQUEST-type output within the current topic, current emotional content and current topic properties.
- A STATEMENT-type output within the current topic, current emotional content and current topic properties.

instance, if the emotion "joy" has a high value, then the corresponding antagonistic emotion "sadness" may be assigned a low value.

The diagram of a generalized emotions control mechanism is shown in FIG.14. The emotional state of the system implemented in the form of an application, for example, for the Internet, may be influenced by its past emotions (i.e., if it has been happy in the last few dialogues) or system actions *S*, a user's phrase *U* (for instance if he/she is abusive), and external data W (such as a large change in stock prices in case of a portfolio-specialization). These elements are shown as a part of the external input step 170 in FIG. 14. The state of the system may be defined by the set of emotions and other variables such as the context of the conversation *M*, information about which is sent by the meta-control mechanism 14 and the discourse control mechanism 15 (step 171). The state of the system is transformed into a new one under the influence of inputs which results in the activation level change of the primary emotions (step 172). The effect of the past context and emotions is covered by the feedback step 173. The functions $F[\varepsilon(k),I(k)]$, $G[\varepsilon(k)]$, $F_d[\varepsilon(k)]$ are the internal rules of the mechanism and in a particular case of implementation are decomposed into the first order differential equation representing the evolution of emotional states E_i:

$$dE/dt = f_U(D_U) + f_S(D_S) + f_M(D_M) + f_d(E)$$

where E and D are multidimensional vectors, and all f are functions of the corresponding parameters.

One of the considered rules specifies the internal dynamics of the states, such as decay values over time (f_d) for a specific emotion when it is rarely activated. If emotion E_1 has a high value at a specified time t, then its values fall over time while there is no persistent input stimulus to drive the system to emotion E_2 as a result of the equation containing f_S. The emotional state of the system is updated every time a user phrase is received for processing and every time the system answers a user phrase, but the emotional state of the system while it is answering is built to be similar to the user's emotional state. The initial (zero) emotional state of the system is referred to as the neutral state. It should be noted that the exact view of the rules for processing the emotional content could be customized, because initially they are defined by scalar functions. This ability allows providing a more precise understanding of a human emotion and changing the content of knowledge bases for different applications without strong efforts from a knowledge engineer. In an uncustomized state, the emotions control mechanism 17 employs a specialized, built-in knowledge base for analyzing the emotional substance of every user input.

2. Meta-actions that are directives or internal commands for the meta-control mechanism 14 in order to change the current flow of a conversation. The list of possible commands corresponds to the set of script (output) intentions. Through the meta-actions a script can redirect the meta-control mechanism to select alternative actions or topics. The Meta actions can be initiated directly from the associated actions base 11 or by the goal control mechanism. In the first case the intention of an action and its parameters are defined in the corresponding Script. The second case relates to a more complicated behavior of the whole system. The dynamics of the process is defined by a periodical call of the goal control mechanism 16 with the help of a query about the active goals available. If the mechanism "decides" that a certain goal must be introduced but the state of this goal is marked as completed, then it generates a meta-action and sends it to the meta-control mechanism 14 which initiates a new search of the best system output.

The actions retrieval mechanism 18 delivers preliminary responses of the system from the meta-control mechanism 14, for example, in the form of a text, references to web pages, javascript actions, animations, sound, video sequences, run-time results of data input from external sources and so on, to the output interface 19. At the last step the action retrieval mechanism 18 receives an appropriate Script from the meta-control mechanism 14 and retrieves the corresponding actions of the system 21 from the associated actions base 11 and through the output interface 19 delivers the answer of the system to the user. At the same time once chosen a script is marked as a used one and can be chosen again provided all the scripts matching the system's response are also marked as used.

Once the script has been derived by the meta-control mechanism 14, the actions retrieval mechanism 18 retrieves a set of appropriate actions from the associated actions base 11. The mechanism also provides the meta-control mechanism 14 with the feedback information, describing what the system is saying or doing. This mechanism uses the following strategies for the subsequent processing of the chosen set of actions:

- Finding the script that matches the specification defined by the meta-control mechanism 14. If there are a variety of scripts matching the specifications then one of them is chosen at random.
- Keeping track of the previously used scripts.

The process of information collection and transformation in this module is shown in FIG. 16. While the topic and the script of the system action are being passed to the action retrieval mechanism 18 through steps 181, 182, the mechanism itself carries out an action or